What is claimed is:

1. A method of generating program source code to perform a mapping task in which enterprise system nested array object fields and legacy system nested array object fields are mapped to one another, said method comprising:

performing a depth-first traversal of a logical tree having a root node, a leaf node for each desired mapping connection, and intermediate nodes between said root node and said leaf nodes, each intermediate node being associated with an array, the depth-first traversal comprising:

(i) for each intermediate node visited when traversing away from said root node, generating program source code to open a loop;

(ii) for each visited leaf node, generating program source code to create the mapping connection represented by said visited leaf node; and

(iii) for each intermediate node having no unvisited children that is visited when traversing towards said root node, generating program source code to close said loop.

2. The method of claim 1, wherein said array with which each intermediate node is associated is an enterprise system array.

3. The method of claim 1, wherein said array with which each intermediate node is associated is a legacy system array.

4. The method of claim 1, wherein said loop opened in step said (i) has a number of iterations corresponding to a size of the array associated with said visited intermediate node.

5. A method of generating program source code to perform a mapping task in which enterprise system nested array object fields and legacy system nested array object fields are mapped to one another, said method comprising:

(a) receiving a list of desired mapping connections between enterprise system fields and legacy system fields;

(b) for each desired connection between an enterprise system field and a legacy system field, determining connection information comprising:

(i) the identity of the nested enterprise arrays containing said enterprise system field;

(ii) the identity of the nested legacy arrays containing said legacy system field; and

(iii) a nesting level of said connection;

(c) creating a logical tree representative of said mapping task comprising:

(i) a root node;

(ii) one leaf node for each said desired mapping connection; and

(iii) for each leaf node, N intermediate nodes interconnecting said leaf node and said root node, where N is equivalent to the determined nesting level of the connection associated with said leaf node, and where each of the N intermediate nodes that is successively further from the root node is associated with an array that is successively more deeply nested; and

(d) performing a depth-first traversal of said logical tree to generate mapping source code, said traversal comprising:

(i) for each intermediate node visited when traversing away from said root node, generating program source code to open a loop;

(ii) for each visited leaf node, generating program source code to create the mapping connection represented by said visited leaf node; and

(iii) for each intermediate node having no unvisited children that is visited when traversing towards said root node, generating program source code to close said loop.

6. The method of claim 5, wherein said array with which said each of the N intermediate nodes is associated is an enterprise system array containing an enterprise system field to be mapped.

7. The method of claim 5, wherein said array with which said each of the N intermediate nodes is associated is a legacy system array containing a legacy system field to be mapped.

8. The method of claim 5, wherein said loop opened in said step (d) substep (i) has a number of iterations corresponding to a size of the array associated with said visited intermediate node.

9. The method of claim 5, wherein said step (c) further comprises, for each said leaf node corresponding to a desired connection, associating with said leaf node at least some of the connection information determined in said step (b).

10. The method of claim 5, wherein said connection information determined in said step (b) for each desired mapping connection further comprises a mapping directionality indicator to indicate whether said connection permits a reading of said legacy system field into said enterprise system field, a writing of said enterprise system field into said legacy system field, or both.

11. The method of claim 10, wherein said step (c) further comprises, for each said leaf

node corresponding to a desired connection, associating with said leaf node a determined mapping directionality indicator.

12. The method of claim 1, wherein said logical tree is represented by a tree data structure.

13. A computer readable medium storing computer software that, when loaded into a computing device, adapts said device to:

perform a depth-first traversal of a logical tree having a root node, a leaf node for each desired mapping connection, and intermediate nodes between said root node and said leaf nodes, each intermediate node being associated with an array, the depth-first traversal comprising:

(i) for each intermediate node visited when traversing away from said root node, generating program source code to open a loop;

(ii) for each visited leaf node, generating program source code to create the mapping connection represented by said visited leaf node; and

(iii) for each intermediate node having no unvisited children that is visited when traversing towards said root node, generating program source code to close said loop.

14. The medium of claim 13, wherein said array with which each intermediate node is associated is an enterprise system array.

15. The medium of claim 13, wherein said array with which each intermediate node is associated is a legacy system array.

16. The medium of claim 13, wherein the opened loop has a number of iterations corresponding to a size of the array associated with said visited intermediate node.

17. A computer readable medium storing computer software that, when loaded into a computing device, adapts said device to:

(a) receive a list of desired mapping connections between enterprise system fields and legacy system fields;

(b) determine, for each desired connection between an enterprise system field and a legacy system field, connection information comprising:

(i) the identity of the nested enterprise arrays containing said enterprise system field;

(ii) the identity of the nested legacy arrays containing said legacy system field; and

(iii) a nesting level of said connection;

(c) create a logical tree representative of said mapping task comprising:

(i) a root node;

(ii) one leaf node for each said desired mapping connection; and

(iii) for each leaf node, N intermediate nodes interconnecting said leaf node and said root node, where N is equivalent to the determined nesting level of the connection associated with said leaf node, and where each of the N intermediate nodes that is successively further from the root node is associated with an array that is successively more deeply nested; and

(d) perform a depth-first traversal of said logical tree to generate mapping source code, said traversal comprising:

(i) for each intermediate node visited when traversing away from said root node, generating program source code to open a loop;

(ii) for each visited leaf node, generating program source code to create the mapping connection represented by said visited leaf node; and

(iii) for each intermediate node having no unvisited children that is visited when traversing towards said root node, generating program source code to close said loop.

18. The medium of claim 17, wherein said array with which said each of the N intermediate nodes is associated is an enterprise system array containing an enterprise system field to be mapped.

19. The medium of claim 17, wherein said array with which said each of the N intermediate nodes is associated is a legacy system array containing a legacy system field to be mapped.

20. The medium of claim 17, wherein said loop opened in step (d)(i) has a number of iterations corresponding to a size of the array associated with said visited intermediate node.

21. The medium of claim 17, wherein said element (c) to create a logical tree further comprises, for each said leaf node corresponding to a desired connection, associating with said leaf node at least some of the connection information determined in said element (b).

22. The medium of claim 17, wherein said connection information determined in said element (b) for each desired mapping connection further comprises a mapping directionality indicator to indicate whether said connection permits a reading of said legacy system field into said enterprise system field, a writing of said enterprise system field into said legacy system field, or both.

23. The medium of claim 22, wherein said element (c) further comprises, for each said leaf node corresponding to a desired connection, associating with said leaf node a determined mapping directionality indicator.

24. In a computing environment including a processor and persistent storage memory in communication with said processor, a system for performing a depth-first traversal of a logical tree having a root node, a leaf node for each desired mapping connection, and intermediate nodes between said root node and said leaf nodes, each intermediate node being associated with an array, said system comprising:

means, for each intermediate node visited when traversing away from said root node, for generating program source code to open a loop;

means, for each visited leaf node, for generating program source code to create the mapping connection represented by said visited leaf node; and

means, for each intermediate node having no unvisited children that is visited when traversing towards said root node, for generating program source code to close said loop.

25. In a computing environment including a processor and persistent storage memory in communication with said processor, a system comprising:

(a) means for receiving a list of desired mapping connections between enterprise system fields and legacy system fields;

(b) means for determining, for each desired connection between an enterprise system field and a legacy system field, connection information comprising:

(i) the identity of the nested enterprise arrays containing said enterprise system field;

(ii) the identity of the nested legacy arrays containing said legacy system field; and

(iii) a nesting level of said connection;

(c) means for creating a logical tree representative of said mapping task comprising:

(i) a root node;

(ii) one leaf node for each said desired mapping connection; and

(iii) for each leaf node, N intermediate nodes interconnecting said leaf node and said root node, where N is equivalent to the determined nesting level of the connection associated with said leaf node, and where each of the N intermediate nodes that is successively further from the root node is associated with an array that is successively more deeply nested; and

(d) means for performing a depth-first traversal of said logical tree to generate mapping source code, said traversal comprising:

(i) for each intermediate node visited when traversing away from said root node, means for generating program source code to open a loop;

(ii) for each visited leaf node, means for generating program source code to create the mapping connection represented by said visited leaf node; and

(iii) for each intermediate node having no unvisited children that is visited when traversing towards said root node, means for generating program source code to close said loop.

26. A method of generating program source code to perform a mapping task in which nested array object fields of a first system and nested array object fields of a second system are mapped to one another, said method comprising:

performing a depth-first traversal of a logical tree having a root node, a leaf node for each desired mapping connection, and intermediate nodes between said root node and said leaf nodes, each intermediate node being associated with an array, the depth-first traversal comprising:

(i) for each intermediate node visited when traversing away from said root node, generating program source code to open a loop;

(ii) for each visited leaf node, generating program source code to create the mapping connection represented by said visited leaf node; and

(iii) for each intermediate node having no unvisited children that is visited when traversing towards said root node, generating program source code to close said loop.

27. A method of generating program source code to perform a mapping task in which nested array object fields of a first system and nested array object fields of a second system are mapped to one another, said method comprising:

(a) receiving a list of desired mapping connections between nested array object fields of a first system and nested array object fields of a second system;

(b) determining, for each desired connection between a field of the first system and a field of the second system, connection information comprising:

(i) the identity of the nested arrays containing said first system field;

(ii) the identity of the nested arrays containing said second system field; and

(iii) a nesting level of said connection;

(c) creating a logical tree representative of said mapping task comprising:

(i) a root node;

(ii) one leaf node for each said desired mapping connection; and

(iii) for each leaf node, N intermediate nodes interconnecting said leaf node and said root node, where N is equivalent to the determined nesting level of the connection associated with said leaf node, and where each of the N intermediate nodes that is successively further from the root node is associated with an array that is successively more deeply nested; and

(d) performing a depth-first traversal of said logical tree to generate mapping source code, said traversal comprising:

(i) for each intermediate node visited when traversing away from said root node, generating program source code to open a loop;

(ii) for each visited leaf node, generating program source code to create the mapping connection represented by said visited leaf node; and

(iii) for each intermediate node having no unvisited children that is visited when traversing towards said root node, generating program source code to close said loop.